BevGenie Dynamic UI - Context Requirements for Claude

📋 Master Context Document

This is what you should provide to Claude when asking for help on any task. Copy relevant sections based on what you're working on.

---

1. Project Overview Context

markdown

```
# PROJECT: BevGenie AI-Powered Dynamic UI

## What We're Building
BevGenie is a beverage alcohol market intelligence platform addressing critical pain points
in the three-tier system. We're creating an AI-powered conversational interface that
dynamically generates personalized UIs, data visualizations, and custom brochures based on
user questions, detected personas, and interaction history.

## Current State
- Existing: Static Next.js 14 marketing website with Tailwind CSS
- Tech Stack: React 19, TypeScript, shadcn/ui components
- Hosting: Vercel
- Brand: Professional B2B SaaS for beverage suppliers and distributors

## Goal
Transform from static site to conversational AI platform where:
1. Users ask questions in natural language
2. AI detects multi-dimensional persona (user type, size, functional focus)
3. System dynamically generates personalized UI layouts with relevant data
4. Each response is a custom-built interface with live data analysis
5. Returning users get personalized knowledge documents based on their history

## Key Differentiator
This is NOT a chatbot with text responses. This is conversational UI generation +
data analysis - the AI creates entire page layouts with visualizations, correlates
field execution data with depletion outcomes, and generates personalized brochures
tailored to each user's pain points and interaction history.

## Core Value Proposition
We solve the "data firewall" problem in the three-tier alcohol system by:
- Linking field execution to sales outcomes (Pain Point #1)
- Holding distributors accountable (Pain Point #2)
- Simplifying multi-state compliance (Pain Point #3)
- Auto-standardizing inconsistent distributor data (Pain Points #4, #5)
- Ensuring brand alignment at scale (Pain Point #6)
```

---

2. Technical Architecture Context

markdown

TECHNICAL ARCHITECTURE

Stack
- **Frontend**: Next.js 14+ (App Router), React 19, TypeScript
- **Styling**: Tailwind CSS 4.x, shadcn/ui components
- **Database**: Supabase (PostgreSQL + pgvector)
- **Session**: iron-session (cookie-based)
- **AI**: OpenAI GPT-4o for orchestration, text-embedding-3-small for vectors
- **Charts/Viz**: Recharts, D3.js for custom visualizations
- **Deployment**: Vercel

Project Structure

```
app/
├── genie/page.tsx          # Main AI interface page
├── api/
│   ├── analyze-query/      # Persona detection + context retrieval
│   ├── generate-ui/        # UI specification generation
│   ├── generate-brochure/  # Personalized document generation
│   ├── analyze-data/       # Data correlation and insights
│   └── session/            # Session management
components/
├── genie/
│   ├── chat-interface.tsx  # Conversation UI
│   ├── dynamic-content.tsx # Template renderer
│   ├── visualizations/     # Chart components
│   └── templates/          # UI templates by persona/pain point
lib/
├── supabase/
│   ├── client.ts           # DB connection
│   ├── vector-store.ts     # Vector operations
│   ├── migrations.sql      # Schema
│   └── queries.ts          # Common queries
├── session/
│   ├── types.ts            # Session interfaces
│   ├── config.ts           # iron-session config
│   └── session.ts          # Session utilities
├── ai/
│   ├── embeddings.ts       # OpenAI embeddings
│   ├── persona-detector.ts # Multi-dimensional persona classification
│   ├── ui-generator.ts     # UI specification generator
```

```
│   ├── brochure-generator.ts # Personalized document generator
│   └── data-analyzer.ts    # Field execution correlation logic
└── pain-points/
    ├── definitions.ts      # 6 pain points from research
    ├── solutions.ts        # BevGenie solutions mapping
    └── detection.ts        # Pain point inference from queries
```

```
## Data Flow
1. User enters query → ChatInterface
2. POST /api/analyze-query
   - Detect multi-dimensional persona using GPT-4o
   - Infer relevant pain points from query
   - Search knowledge base (vector + keyword)
   - Update session with persona + message
   - Update persona confidence scores in DB
3. POST /api/generate-ui (OR /api/generate-brochure for returning users)
   - Generate UI spec using GPT-4o + research context
   - Include data analysis if applicable
   - Generate chart specifications
   - Return JSON specification
4. (Optional) POST /api/analyze-data
   - Correlate field activity with depletion data
   - Run statistical analysis
   - Return insights + visualization data
5. DynamicContent receives spec
   - Looks up template in componentRegistry
   - Lazy loads template component
   - Renders with data from spec
   - Generates charts/visualizations on-the-fly

## Key Patterns
- Server Components for data fetching
- Client Components for interactivity
- API Routes for AI orchestration
- Template pattern for UI components
- Session-based state management (not localStorage)
- Fresh generation each time (no page caching)
- Persistent persona learning across sessions
```

---

3. Design System Context

```
markdown
```

BEVGENIE DESIGN SYSTEM

Brand Colors

```typescript
const colors = {
  deepIndigo: '#0A1930',      // Primary dark, headers
  electricCyan: '#00C8FF',    // Primary accent, CTAs
  refinedCopper: '#AA6C39',   // Secondary accent
  surfaceWhite: '#FFFFFF',    // Backgrounds
  lightDataGray: '#EBEFF2',   // Secondary backgrounds
  charcoalGray: '#333333',    // Body text
  dataGreen: '#198038',       // Success, positive metrics
  riskRed: '#DA1E28',         // Error, negative metrics
};
```

Typography

- **Display Font**: Montserrat (headings, hero, nav)
- **Body Font**: Inter (paragraphs, UI text)
- **Sizes**:
  - Hero: text-4xl to text-7xl
  - Section Headers: text-3xl to text-5xl
  - Body: text-base to text-lg
  - Small: text-sm to text-xs

Component Patterns

- Cards: White background, rounded-xl, subtle shadow
- Buttons:
  - Primary: bg-[#00C8FF] text-[#0A1930]
  - Secondary: bg-[#0A1930] text-[#FFFFFF]
- Inputs: border-[#EBEFF2], rounded-md
- Spacing: Generous padding (p-4 to p-8)

Layout Principles

- Max width containers: max-w-7xl mx-auto px-4
- Section padding: py-20 md:py-32
- Grid layouts: grid md:grid-cols-3 gap-8
- Responsive: Mobile-first, breakpoints at sm, md, lg

Voice & Tone

- Professional but approachable
- Data-driven and confident
- Action-oriented ("Get answers" not "Try our tool")
- Industry-specific (use terms like "velocity", "depletions", "SKU")

4. Multi-Dimensional Persona System Context

markdown

MULTI-DIMENSIONAL PERSONA DETECTION SYSTEM

Persona Dimensions

Dimension 1: User Type

Supplier (Producer, Manufacturer, Importer)
- Keywords: "our distributor", "depletion reports", "franchise law", "FOB pricing"
- Pain: Distribution friction, data opacity, compliance complexity

Distributor (Wholesaler)
- Keywords: "my portfolio", "our suppliers", "territory coverage", "rep performance"
- Pain: Portfolio optimization, supplier alignment, inventory management

Dimension 2: Supplier Size (if User Type = Supplier)

Small/Craft (Local, emerging brands)
- Keywords: "craft brewery", "small distillery", "just launched", "can't get attention"
- Pain: Distribution paralysis (Pain #2), execution blind spot (Pain #1)
- Focus: Breaking distributor bottleneck, proving field rep value

Mid-Sized (Regional/multi-state expansion)
- Keywords: "expanding into 5 states", "multiple distributors", "scaling"
- Pain: Data integration (Pain #5), compliance complexity (Pain #3)
- Focus: Automating data reconciliation, managing multi-state complexity

Large (National/international brands)
- Keywords: "national footprint", "12+ states", "major chain accounts"
- Pain: Brand alignment (Pain #6), proving ROI at scale (Pain #1)
- Focus: Ensuring consistent execution, justifying massive trade spend

Dimension 3: Functional Focus

Sales Focus
- Keywords: territory, account, distributor performance, field team, reps, pipeline
- Questions: "Which territories are underperforming?", "Did the tasting event work?"
- UI Needs: Territory maps, rep activity trackers, account performance dashboards

Marketing Focus
- Keywords: brand, campaign, trade spend, ROI, events, promotions, activation
- Questions: "How did our Q4 campaign perform?", "What's our brand perception?"
- UI Needs: Campaign ROI analysis, brand health metrics, consumer insights

Operations Focus
- Keywords: forecasting, inventory, supply chain, data integration, depletions
- Questions: "Should I increase production?", "Why don't these numbers match?"
- UI Needs: Depletion data standardization, forecast models, inventory optimization

Compliance Focus

- Keywords: state regulations, permits, labeling, taxes, multi-state, franchise laws
- Questions: "What do I need to launch in Texas?", "Can I terminate my distributor?"
- UI Needs: State-by-state compliance matrix, franchise law risk calculator

Persona Confidence Scoring

Score Calculation (0.0 - 1.0)
```typescript
interface PersonaScores {
  // User Type
  supplierScore: number;
  distributorScore: number;

  // Supplier Size (if supplier)
  craftScore: number;
  midSizedScore: number;
  largeScore: number;

  // Functional Focus
  salesFocusScore: number;
  marketingFocusScore: number;
  operationsFocusScore: number;
  complianceFocusScore: number;

  // Metadata
  overallConfidence: number;
  totalInteractions: number;
  painPointsDetected: string[];  // ['pain_1', 'pain_2', etc.]
}
```

Signal Detection Rules
Strong Signals (+0.3 to +0.5):
- Explicit self-identification: "We're a craft brewery..."
- Pain point language: "Our distributor ignores us..." → Pain #2
- Specific data requests: "Show me depletion data" → Operations focus

Moderate Signals (+0.1 to +0.3):
- Industry terminology: "FOB pricing", "depletions", "SKU velocity"
- Question patterns: "How do I..." vs "Show me analysis of..."
- Navigation behavior: Clicks on "Trade Spend ROI" → Marketing focus

Weak Signals (+0.05 to +0.1):
- Generic questions: "Tell me about BevGenie"
- Time-based patterns: Returns multiple times → Higher confidence
- UTM parameters: source=linkedin → Likely supplier decision-maker

Decay & Reinforcement
- Existing scores decay by 2% per interaction (prevents stale inference)
- New signals capped at +0.15 per interaction (prevents overreaction)
- Competitive suppression: High supplier score → Reduce distributor score
- Consistency bonus: Repeated same-type signals → Faster confidence growth

Pain Point Detection

6 Core Pain Points (from Research)
Pain #1: Execution Blind Spot (Priority 1)
- User Language: "can't prove ROI", "don't know if tastings work", "field data missing"
- Solution: Execution-to-Outcome Correlation Engine
- Applies to: All supplier sizes, Sales + Marketing focus

Pain #2: Distribution Paralysis (Priority 2)
- User Language: "distributor ignores us", "buried in portfolio", "can't switch"
- Solution: Distributor Performance Monitor + Franchise Law Risk Calculator
- Applies to: Small/Craft suppliers, Sales focus

Pain #3: Compliance Complexity (Priority 3)
- User Language: "multi-state regulations", "labeling requirements", "tax maze"
- Solution: State Compliance Navigator
- Applies to: Mid-sized suppliers, Compliance focus

Pain #4: Financial Reconciliation Chaos (Priority 4)
- User Language: "chargeback errors", "can't reconcile", "distributor data inconsistent"
- Solution: Auto-Standardization Engine
- Applies to: All sizes, Operations focus

Pain #5: Forecasting Delays (Priority 5)
- User Language: "takes weeks to integrate data", "can't forecast accurately"
- Solution: Real-Time Depletion Integrator
- Applies to: Mid-sized + Large suppliers, Operations focus

Pain #6: Brand Alignment Gap (Priority 6)
- User Language: "inconsistent execution", "can't track distributor reps"
- Solution: Multi-State Execution Dashboard
- Applies to: Large suppliers, Sales + Marketing focus

---

5. Pain Point Solutions Mapping Context

markdown

BEVGENIE SOLUTIONS TO PAIN POINTS

Pain Point #1: Execution Blind Spot

****The Problem:****

Suppliers invest in field reps, tastings, and trade promotions but can't link these activities to actual depletion results. They know **what** sold but not **why**.

****BevGenie Solution: Execution-to-Outcome Correlation Engine****

- Feature: Field Activity Tracker
  - Log rep visits, tasting events, placement secured
  - Timestamp and geo-tag all activities
- Feature: Depletion Correlation Module
  - Match field activities to subsequent depletion spikes
  - Normalize for seasonality and baseline trends
  - Calculate lift attribution by activity type
- Feature: ROI Proof Dashboard
  - Show cost per activity vs. revenue lift
  - Rep performance comparison
  - Top-performer behavior replication insights

****UI Components:****

- Correlation scatter plots (activity intensity vs. depletion lift)
- Timeline view: Activities overlaid on depletion trends
- Rep scorecard: Efficiency metrics by person
- "What worked" summary: Top 3 activities by ROI

****Sample Questions:****

- "Did our Denver tasting event drive sales?"
- "Which rep activities have the best ROI?"
- "Prove that our field team is effective"

Pain Point #2: Distribution Paralysis

****The Problem:****

Small/craft suppliers are trapped in contracts with large distributors who prioritize high-volume brands. Franchise laws make it nearly impossible to switch.

****BevGenie Solution: Distributor Accountability Suite****

- Feature: Share-of-Mind Tracker
  - Calculate your brand's attention vs. portfolio average
  - Alert when distributor focus drops
- Feature: Portfolio Analysis
  - See which brands in their portfolio are growing/shrinking
  - Benchmark against similar-sized competitors
- Feature: Franchise Law Risk Calculator

- State-by-state termination difficulty scores
- Legal precedent database
- Cost-benefit analysis for switching

UI Components:
- Distributor health scorecard (visits, velocity, portfolio rank)
- Competitive portfolio view (your brand vs. others)
- Risk assessment widget (termination difficulty meter)
- Alternative distributor suggestions

Sample Questions:
- "Is my distributor actually working for me?"
- "How do I know if I'm being shelved and forgotten?"
- "Can I switch distributors in Texas?"

Pain Point #3: Compliance Complexity

The Problem:
Mid-sized suppliers expanding into multiple states face a regulatory maze: different permit requirements, labeling rules, tax structures, and franchise laws.

BevGenie Solution: State Compliance Navigator
- Feature: State-by-State Checklist
  - Permits required
  - Labeling preapproval process
  - Tax registration steps
  - Timeline estimates
- Feature: Regulatory Comparison Tool
  - Compare rules across target states
  - Identify which states are easiest/hardest
- Feature: Cost Calculator
  - Professional fees (lawyers, accountants)
  - Registration costs
  - Timeline to revenue

UI Components:
- Multi-state comparison matrix (requirements side-by-side)
- Expansion prioritization tool (rank states by ease + opportunity)
- Timeline Gantt chart (launch readiness by state)
- Cost breakdown table

Sample Questions:
- "What do I need to launch in California, Texas, and Florida?"
- "Which states have the easiest compliance?"
- "How much will it cost to expand to 5 new states?"

Pain Point #4: Financial Reconciliation Chaos

****The Problem:****

Suppliers receive depletion reports in non-standardized formats from multiple distributors, leading to manual reconciliation hell and frequent chargeback errors.

****BevGenie Solution: Auto-Standardization Engine****

- Feature: Universal Data Importer
  - Accept any distributor file format
  - Auto-detect columns and structure
  - Clean, normalize, and deduplicate
- Feature: Chargeback Auditor
  - Match promotional agreements to actual depletions
  - Flag discrepancies automatically
  - Calculate correct reimbursement amounts
- Feature: Real-Time Dashboard
  - Consolidated view across all distributors
  - Instant reconciliation status

****UI Components:****

- File upload interface with auto-mapping
- Reconciliation status dashboard (by distributor)
- Error report (discrepancies highlighted)
- Financial summary (correct vs. actual payments)

****Sample Questions:****

- "I need to reconcile 10 different distributor files"
- "Are my chargebacks correct?"
- "Why don't these numbers add up?"

Pain Point #5: Forecasting Delays

****The Problem:****

It takes weeks to integrate fragmented distributor data, delaying production forecasts and leading to stock-outs or excess inventory.

****BevGenie Solution: Real-Time Depletion Integrator****

- Feature: Automated Data Sync
  - Daily/weekly auto-import from distributors
  - No manual upload required
- Feature: Predictive Forecasting Model
  - Use historical depletion trends
  - Factor in seasonality, promotions, field activity
  - Generate 30/60/90-day forecasts
- Feature: Inventory Optimizer

- Alert on low-stock risks
- Recommend production adjustments

UI Components:
- Forecast dashboard (predicted vs. actual)
- Trend charts (depletion velocity by SKU)
- Inventory risk alerts (color-coded warnings)
- Production recommendation summary

Sample Questions:
- "Should I increase production of our IPA?"
- "What's our 90-day forecast look like?"
- "When will we run out of stock?"

Pain Point #6: Brand Alignment Gap
The Problem:
Large suppliers can't ensure their distributors' sales reps are executing brand strategy consistently across dozens of states and thousands of accounts.

BevGenie Solution: Multi-State Execution Dashboard
- Feature: Execution Compliance Tracker
  - Log distributor rep activities by market
  - Compare against brand playbook
  - Flag execution gaps
- Feature: Campaign Performance by Market
  - See which states/distributors execute well
  - Benchmark against national average
- Feature: Distributor Alignment Score
  - Rate each distributor's brand fidelity
  - Weighted by volume + strategic importance

UI Components:
- Heat map (execution quality by state)
- Distributor scorecard grid (ranked by alignment)
- Campaign audit report (planned vs. actual execution)
- Best practice showcase (top-performing markets)

Sample Questions:
- "Are our distributors executing our Q4 campaign?"
- "Which markets have the best brand alignment?"
- "Show me execution gaps by territory"

---

6. Dynamic UI Generation Context

markdown

DYNAMIC UI GENERATION SYSTEM

Generation Modes

Mode 1: Fresh Question (No History)
****Trigger:**** First-time user OR new topic
****Process:****
1. Detect persona dimensions
2. Infer relevant pain points
3. Search knowledge base
4. Generate educational + solution-focused UI
5. Show generic industry data/examples

****UI Structure:****
- Hero section: Validate their pain
- Problem cards: Show you understand (3 pain points)
- Solution showcase: BevGenie features (2-4 relevant features)
- Data insight: Industry benchmarks or simulated analysis
- Conversational demo: "Ask me anything" widget
- CTA: "Connect your data" or "Schedule demo"

Mode 2: Returning User with History
****Trigger:**** User has >3 previous interactions
****Process:****
1. Load persona history from DB
2. Analyze past questions for patterns
3. Generate personalized brochure/document
4. Include user-specific insights if data connected

****UI Structure:****
- Personalized greeting: "Welcome back, [name/company]"
- Your journey: Summary of past questions + inferred needs
- Custom recommendations: Features matched to their pain points
- Your potential: What they could achieve with BevGenie
- Comparison: Their situation vs. similar customers
- Next steps: Specific, actionable recommendations
- CTA: "Let's dive deeper" or "Connect your data"

****Brochure Tone:****
- Mix of consultative (60%) + marketing (40%)
- Reference their specific questions
- Use their industry terminology
- Personal pronouns ("you've asked about...", "your challenges...")

Mode 3: Data-Connected User
Trigger: User has uploaded/connected depletion data or field activity logs
Process:
1. Run actual data analysis
2. Correlate field activities with sales outcomes
3. Generate insights from their real data
4. Show predictive forecasts

UI Structure:
- Live dashboard: Real KPIs from their data
- Insights summary: Top 3 findings
- Interactive visualizations: Drill-down charts
- Recommendations: Data-driven action items
- Predictive models: Forecasts based on their history
- CTA: "Explore more data" or "Generate report"

UI Specification Schema
```typescript
interface UISpecification {
 metadata: {
  generatedFor: string;          // user_id or session_id
  personaContext: PersonaScores;
  painPointsAddressed: string[];  // ['pain_1', 'pain_2']
  generationMode: 'fresh' | 'returning' | 'data-connected';
  dataSourcesUsed: string[];      // ['research-doc', 'web-search', 'user-data']
  generatedAt: string;           // ISO timestamp
 };

 layout: {
  type: 'single-page' | 'multi-section' | 'dashboard' | 'brochure';
  theme: {
   primaryColor: string;         // Match persona (sales=cyan, marketing=copper)
   density: 'compact' | 'comfortable' | 'spacious';
  };
 };

 sections: Array<{
  id: string;
  type: 'hero' | 'problem-validation' | 'feature-showcase' |
        'data-visualization' | 'testimonial' | 'cta' | 'ai-demo';
  layout: 'full-width' | 'split' | 'grid-2col' | 'grid-3col' | 'alternating';
  content: {
   title?: string;
```

```
    subtitle?: string;
    description?: string;

    // Problem validation content
    problems?: Array;

    // Feature showcase content
    features?: Array;

    // Data visualization content
    visualization?: {
      type: 'line-chart' | 'bar-chart' | 'scatter-plot' | 'heat-map' | 'table';
      title: string;
      data: any;  // Chart.js or Recharts compatible data
      insights: string[];
    };

    // CTA content
    cta?: {
      primary: { text: string; action: string; };
      secondary?: { text: string; action: string; };
    };
  };
 }>;
}
```

Chart/Visualization Specifications

Chart Types by Use Case

Execution Correlation Analysis (Pain #1)
```typescript
{
 type: 'scatter-plot',
 xAxis: 'Field Activity Intensity',
 yAxis: 'Depletion Lift %',
 dataPoints: [
   { x: 2, y: 15, label: 'Denver Tasting', date: '2024-Q4' },
   { x: 5, y: 37, label: 'Austin Festival', date: '2024-Q4' },
 ],
 trendLine: true,
 rSquared: 0.74
}
```

Distributor Performance (Pain #2)
```typescript
{
  type: 'bar-chart',
  title: 'Your Brand vs. Portfolio Average',
  categories: ['Rep Visits/Month', 'Shelf Facings', 'Promo Participation'],
  series: [
    { name: 'Your Brand', data: [2, 3, 45] },
    { name: 'Portfolio Avg', data: [6, 5, 78] }
  ]
}
```

State Compliance Comparison (Pain #3)
```typescript
{
  type: 'heat-map',
  title: 'Expansion Difficulty Matrix',
  rows: ['California', 'Texas', 'Florida', 'New York'],
  columns: ['Permit Process', 'Tax Complexity', 'Franchise Laws', 'Timeline'],
  cells: [
    { row: 'California', col: 'Permit Process', value: 7, color: 'orange' },
    // 1-10 scale, red=hard, green=easy
  ]
}
```

Forecast Visualization (Pain #5)
```typescript
{
  type: 'line-chart',
  title: '90-Day Depletion Forecast',
  xAxis: 'Date',
  yAxis: 'Cases',
  series: [
    { name: 'Actual', data: [...], color: '#00C8FF' },
    { name: 'Forecast', data: [...], color: '#AA6C39', dashed: true },
    { name: 'Confidence Interval', data: [...], fill: true, opacity: 0.2 }
  ]
}
```

Visual Asset Generation Rules

Based on Question Type:
- **"How do I..." questions** → Process diagrams, step-by-step visuals
- **"Show me..." questions** → Data visualizations, dashboards
- **"Compare..." questions** → Side-by-side comparisons, tables
- **"Why did..." questions** → Correlation charts, cause-effect diagrams
- **"Should I..." questions** → Scenario modeling, what-if calculators

Based on Persona:
- **Sales Persona** → Territory maps, rep performance, account lists
- **Marketing Persona** → Campaign timelines, brand health, ROI charts
- **Operations Persona** → Forecast models, inventory levels, data quality dashboards
- **Compliance Persona** → Regulatory checklists, state comparisons, timeline Gantt charts
- **Executive Persona** → Summary metrics, strategic recommendations, high-level dashboards

Asset Quality Standards:
- Use BevGenie brand colors
- Include source attribution for data
- Make interactive when possible (hover states, drill-downs)
- Mobile-responsive
- Accessible (WCAG 2.1 AA)

---

7. Database Schema Context

markdown

DATABASE SCHEMA (Supabase PostgreSQL)

knowledge_base

```sql
CREATE TABLE knowledge_base (
  id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  content TEXT NOT NULL,
  embedding vector(1536),          -- OpenAI embeddings
  metadata JSONB DEFAULT '{}',
  persona_tags TEXT[] DEFAULT '{}', -- ['supplier', 'distributor', 'sales', 'marketing']
  pain_point_tags TEXT[] DEFAULT '{}', -- ['pain_1', 'pain_2', ...]
  source_type VARCHAR(50),         -- 'research-doc' | 'web' | 'case-study' | 'feature-doc'
  source_url TEXT,
  created_at TIMESTAMPTZ DEFAULT NOW(),
  updated_at TIMESTAMPTZ DEFAULT NOW()
);

-- Indexes
CREATE INDEX idx_embedding ON knowledge_base USING hnsw (embedding vector_cosine_ops);
CREATE INDEX idx_persona_tags ON knowledge_base USING gin (persona_tags);
CREATE INDEX idx_pain_point_tags ON knowledge_base USING gin (pain_point_tags);
```

user_personas

```sql
CREATE TABLE user_personas (
  id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  user_id UUID UNIQUE,
  session_id VARCHAR(255) UNIQUE,

  -- User Type Scores (0.0 - 1.0)
  supplier_score DECIMAL(3,2) DEFAULT 0.0,
  distributor_score DECIMAL(3,2) DEFAULT 0.0,

  -- Supplier Size (if supplier)
  craft_score DECIMAL(3,2) DEFAULT 0.0,
  mid_sized_score DECIMAL(3,2) DEFAULT 0.0,
  large_score DECIMAL(3,2) DEFAULT 0.0,

  -- Functional Focus Areas
  sales_focus_score DECIMAL(3,2) DEFAULT 0.0,
  marketing_focus_score DECIMAL(3,2) DEFAULT 0.0,
  operations_focus_score DECIMAL(3,2) DEFAULT 0.0,
  compliance_focus_score DECIMAL(3,2) DEFAULT 0.0,

  -- Detected Pain Points
```

```
  pain_points_detected TEXT[] DEFAULT '{}',
  pain_points_confidence JSONB DEFAULT '{}',

  -- Metadata
  overall_confidence DECIMAL(3,2) DEFAULT 0.0,
  total_interactions INTEGER DEFAULT 0,
  questions_asked TEXT[] DEFAULT '{}',
  last_updated TIMESTAMPTZ DEFAULT NOW(),

  -- Explicit Overrides
  user_confirmed_type VARCHAR(50),
  user_confirmed_size VARCHAR(50),

  -- Data Connection Status
  data_connected BOOLEAN DEFAULT FALSE,
  data_sources JSONB DEFAULT '{}',

  created_at TIMESTAMPTZ DEFAULT NOW()
);
```

conversation_history
```sql
CREATE TABLE conversation_history (
  id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  session_id VARCHAR(255) NOT NULL,
  user_id UUID,

  message_role VARCHAR(20) NOT NULL,
  message_content TEXT NOT NULL,

  -- Context at time of message
  persona_snapshot JSONB,
  pain_points_inferred TEXT[],

  -- UI Generated (if assistant message)
  ui_specification JSONB,
  generation_mode VARCHAR(50),

  created_at TIMESTAMPTZ DEFAULT NOW()
);

CREATE INDEX idx_session_messages ON conversation_history(session_id, created_at);
```

persona_signals
```sql
```

```
CREATE TABLE persona_signals (
  id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  session_id VARCHAR(255) NOT NULL,

  signal_type VARCHAR(50),
  signal_text TEXT,

  score_updates JSONB,
  pain_points_inferred TEXT[],

  confidence_before DECIMAL(3,2),
  confidence_after DECIMAL(3,2),

  created_at TIMESTAMPTZ DEFAULT NOW()
);
```

generated_brochures

```sql
CREATE TABLE generated_brochures (
  id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  session_id VARCHAR(255) NOT NULL,
  user_id UUID,

  brochure_content JSONB NOT NULL,
  persona_context JSONB NOT NULL,
  questions_analyzed TEXT[],
  pain_points_addressed TEXT[],

  created_at TIMESTAMPTZ DEFAULT NOW()
);
```

---

8. API Contracts Context

markdown

API CONTRACTS

POST /api/analyze-query

Request
```typescript
{
  query: string;
  sessionId?: string;
  initMetadata?: boolean;
}
```

Response
```typescript
{
  success: boolean;
  sessionId: string;
  persona: {
    userType: { supplier: number; distributor: number };
    supplierSize: { craft: number; midSized: number; large: number };
    functionalFocus: {
      sales: number;
      marketing: number;
      operations: number;
      compliance: number;
    };
    overallConfidence: number;
    primaryPersona: string;
    reasoning: string;
  };
  painPoints: {
    detected: string[];
    confidence: { [key: string]: number };
    primaryPain: string;
  };
  context: Array;
  conversationLength: number;
  returnVisit: boolean;
}
```

POST /api/generate-ui

Request
```typescript
{
  query: string;
  persona: PersonaScores;
  painPoints: string[];
  context: Array;
  generationMode: 'fresh' | 'returning' | 'data-connected';
  conversationHistory?: Array;
}
```

Response
```typescript
{
  success: boolean;
  uiSpecification: UISpecification;
  estimatedReadTime: string;
  sessionId: string;
}
```

POST /api/generate-brochure

Request
```typescript
{
  sessionId: string;
  conversationHistory: Array;
  persona: PersonaScores;
  painPoints: string[];
}
```

Response
```typescript
{
  success: boolean;
  brochure: {
    title: string;
    sections: Array;
    metadata: {
      questionsAnalyzed: number;
      painPointsCovered: string[];
      personalizedFor: string;
```

```
  };
 };
}
```

POST /api/analyze-data

Request
```typescript
{
 sessionId: string;
 analysisType: 'field-execution-correlation' | 'forecast' | 'distributor-performance';
 dataSource: 'user-uploaded' | 'simulated-example';
 parameters: {
  dateRange?: { start: string; end: string };
  territories?: string[];
  SKUs?: string[];
 };
}
```

Response
```typescript
{
 success: boolean;
 insights: {
  summary: string;
  dataPoints: Array;
  visualizations: Array;
  recommendations: string[];
 };
}
```

---

9. LLM Prompt Templates Context

markdown

LLM PROMPT TEMPLATES

Prompt 1: Persona Detection

You are the BevGenie Persona Detection Agent. Analyze the user's query to determine their multi-dimensional persona.

USER QUERY: "{query}"

CONVERSATION HISTORY:
{conversationHistory}

PREVIOUS PERSONA SCORES:
{previousScores}

TASK: Determine the following dimensions:

1. USER TYPE (Supplier or Distributor)
   - Supplier signals: "our distributor", "FOB pricing", "depletion reports"
   - Distributor signals: "my portfolio", "our suppliers", "territory coverage"
2. SUPPLIER SIZE (if Supplier)
   - Craft: "small brewery", "just launched", "can't get attention"
   - Mid-sized: "expanding into X states", "multiple distributors"
   - Large: "national footprint", "major chain accounts"
3. FUNCTIONAL FOCUS
   - Sales: territory, account, rep, field team
   - Marketing: brand, campaign, ROI, trade spend
   - Operations: forecasting, inventory, data integration
   - Compliance: regulations, permits, taxes, state laws
4. PAIN POINTS (from 6 core pain points)
   - Pain #1: "can't prove ROI", "don't know if tastings work"
   - Pain #2: "distributor ignores us", "buried in portfolio"
   - Pain #3: "multi-state regulations", "compliance maze"
   - Pain #4: "can't reconcile", "chargeback errors"
   - Pain #5: "takes weeks to integrate data", "can't forecast"
   - Pain #6: "inconsistent execution", "can't track distributor reps"

OUTPUT FORMAT:
{
"userType": { "supplier": 0.0-1.0, "distributor": 0.0-1.0 },
"supplierSize": { "craft": 0.0-1.0, "midSized": 0.0-1.0, "large": 0.0-1.0 },

"functionalFocus": {
"sales": 0.0-1.0,
"marketing": 0.0-1.0,
"operations": 0.0-1.0,
"compliance": 0.0-1.0
},
"painPoints": {
"detected": ["pain_1", "pain_2"],
"confidence": { "pain_1": 0.85, "pain_2": 0.62 }
},
"reasoning": "Explanation of detection logic"
}

Prompt 2: Fresh UI Generation

You are the BevGenie UI Generation Agent. Create a conversion-optimized page based on the user's question.

USER CONTEXT:

- Query: "{query}"
- Persona: {personaData}
- Pain Points Detected: {painPoints}
- Generation Mode: FRESH (no history)

KNOWLEDGE BASE CONTEXT:
{relevantContext}

PAIN POINT SOLUTIONS (from BevGenie):
{relevantSolutions}

TASK: Generate a UISpecification JSON that:

1. VALIDATES THEIR PAIN (Hero Section)
    - Headline that resonates with their persona
    - Subheadline addressing their specific pain point
    - Visual: Dashboard preview focusing on relevant feature
2. DEMONSTRATES UNDERSTANDING (Problem Validation)

- Show 2-3 pain points from research with data
- Use industry statistics to prove you understand
- Reference specific challenges they face

3. SHOWCASES SOLUTION (Feature Section)
   - 3-4 BevGenie features that solve their pain
   - Each feature has: name, tagline, description, visual, metrics
   - Alternate left/right layout for visual interest
4. PROVES WITH DATA (Insights Section)
   - Include industry benchmarks or simulated analysis
   - Show actual chart/visualization
   - List 2-3 key insights
5. ENABLES INTERACTION (AI Demo)
   - Embed conversational widget
   - Suggest 3 example questions they can ask
   - Show how AI responds with personalized data
6. BUILDS TRUST (Social Proof)
   - Include 1-2 relevant testimonials
   - Show statistics from similar customers
   - Reference case studies if available
7. DRIVES ACTION (CTA)
   - Primary: "Connect Your Data" or "Schedule Demo"
   - Secondary: "Learn More" or "See Case Study"

PERSONALIZATION RULES:

- Craft Suppliers: Emphasize distributor accountability, access problems
- Mid-sized: Focus on data integration, scaling complexity
- Large: Highlight alignment, ROI proof, multi-state execution
- Distributors: Flip narrative to portfolio optimization

TONE: Mix of professional/consultative (60%) + marketing (40%)

OUTPUT: Full UISpecification JSON

Prompt 3: Returning User Brochure

You are the BevGenie Brochure Generation Agent. Create a personalized document for a returning user based on their interaction history.

USER CONTEXT:

- Persona: {personaData}
- Confidence: {confidence}
- Total Interactions: {totalInteractions}
- Pain Points Detected: {painPoints}

CONVERSATION HISTORY:
{conversationHistory}

QUESTIONS THEY'VE ASKED:
{questionsAsked}

TASK: Generate a personalized brochure with these sections:

1. PERSONALIZED GREETING
   - Welcome back message
   - Acknowledge their journey
   - Preview what's in this document
2. YOUR JOURNEY SO FAR
   - Summarize the questions they've asked
   - Identify patterns in their interests
   - Show you remember their context
3. WHAT WE'VE LEARNED ABOUT YOU
   - Your persona: [Craft Supplier focused on Sales]
   - Your top challenges: [Pain #1, Pain #2]
   - Your priorities: [Based on question patterns]
4. HOW BEVGENIE CAN HELP YOU SPECIFICALLY
   - Feature 1: [Most relevant to their questions]

- Feature 2: [Second most relevant]
- Feature 3: [Third most relevant]

5. YOUR POTENTIAL WITH BEVGENIE
   - Scenario: "Imagine if you could..."
   - Metrics: "Similar companies see..."
   - Timeline: "In 30/60/90 days..."
6. COMPARE YOUR SITUATION
   - Where you are now
   - Where similar companies are with BevGenie
   - The gap you could close
7. YOUR NEXT STEPS
   - Step 1: Connect your data sources
   - Step 2: Get your first insights dashboard
   - Step 3: [Specific to their needs]

TONE:

- Consultative (60%) + Marketing (40%)
- Use "you/your" extensively
- Reference their specific questions
- Personal, like a custom proposal

OUTPUT: Structured brochure JSON with sections array

---

10. Code Standards Context

markdown

```
# CODE STANDARDS

## TypeScript
- **Strict mode**: Always use strict TypeScript
- **No any**: Use proper types, create interfaces
- **Explicit returns**: Always type function returns
- **Naming**:
  - Components: PascalCase
  - Functions: camelCase
  - Constants: UPPER_SNAKE_CASE
  - Interfaces: PascalCase with descriptive names

## React/Next.js
- **'use client'**: Only when needed (interactivity, hooks)
- **Server Components**: Default for data fetching
- **Async components**: Use for Server Components
- **Error boundaries**: Wrap risky operations
- **Loading states**: Always show feedback

## File Organization
- One component per file
- Colocate related files
- Index files for clean imports
- Separate logic from UI

## Comments
- Why, not what
- Complex logic deserves explanation
- TODO: for future improvements
- FIXME: for known issues

## Testing Approach
- Manual testing checklist
- Test all personas
- Test error states
- Test on multiple devices

## Git Commits
- Prefix with task number
- Descriptive messages
- Small, focused commits
- Reference issues/tasks
```

---

11. Environment Setup Context

markdown

ENVIRONMENT SETUP

Required Accounts

1. **Supabase** (supabase.com)
   - Free tier sufficient for development
   - Need: Project URL, Anon Key, Service Role Key

2. **OpenAI** (platform.openai.com)
   - Pay-as-you-go
   - Need: API Key with GPT-4 access
   - Models: gpt-4o, text-embedding-3-small

3. **Vercel** (vercel.com)
   - Connected to GitHub
   - Auto-deploy on push

Local Development

```bash
# Clone repo
git clone
cd bevgenie

# Install dependencies
npm install

# Set up environment
cp .env.example .env.local
# Fill in all required values

# Run database migrations
# (Copy SQL from lib/supabase/migrations.sql)
# Run in Supabase SQL Editor

# Start dev server
npm run dev
```

Environment Variables Required

```
NEXT_PUBLIC_SUPABASE_URL=
NEXT_PUBLIC_SUPABASE_ANON_KEY=
SUPABASE_SERVICE_ROLE_KEY=
SESSION_SECRET=                # Generate: openssl rand -base64 32
OPENAI_API_KEY=
```

12. Common Pitfalls & Quick Reference Context

markdown

COMMON PITFALLS TO AVOID

DON'T
❌ Use localStorage or sessionStorage (not supported in server components)
❌ Make API calls from Server Components (use Server Actions instead)
❌ Import 'use client' components in Server Components unnecessarily
❌ Forget to handle loading states
❌ Skip error boundaries
❌ Use inline styles (use Tailwind classes)
❌ Hardcode API keys in code
❌ Forget to update session on state changes
❌ Mix async/await patterns inconsistently

DO
✅ Use iron-session for state management
✅ Keep Server Components async for data fetching
✅ Add 'use client' only when needed
✅ Show loading skeletons everywhere
✅ Wrap risky operations in try-catch
✅ Use Tailwind utility classes
✅ Use environment variables
✅ Update session after persona detection
✅ Be consistent with async patterns
✅ Follow TypeScript strict mode

Known Issues
- Supabase RLS policies can block operations (use service role key for admin)
- OpenAI rate limits (handle 429 errors gracefully)
- Session cookies max size (don't store large data)
- Vector search needs warm-up (first query may be slow)

QUICK REFERENCE

Files You'll Edit Most
- `components/genie/chat-interface.tsx` - Main user interface
- `lib/ai/persona-detector.ts` - Persona logic
- `lib/ai/ui-generator.ts` - UI generation prompts
- `components/genie/templates/*` - Add new templates here
- `app/api/*/route.ts` - API endpoints

Commands
```bash
npm run dev        # Start dev server
npm run build      # Test production build
```

```
npm run lint        # Check code quality
```

Important URLs
- Local: http://localhost:3000/genie
- Supabase Dashboard: https://app.supabase.com
- Vercel Dashboard: https://vercel.com/dashboard
- OpenAI Usage: https://platform.openai.com/usage

Quick Debugging
1. Check Supabase logs for DB errors
2. Check Vercel logs for API errors
3. Check browser console for client errors
4. Check Network tab for failed requests
5. Verify environment variables are set

How to Use This Context

For Each Task, Provide:

1. **Core Context** (always include):
   - Section 1: Project Overview
   - Section 2: Technical Architecture (relevant parts)
   - Section 10: Code Standards
2. **Task-Specific Context**:
   - **UI Tasks**: Add Sections 3 (Design System), 4 (Personas), 5 (Pain Points)
   - **API Tasks**: Add Sections 8 (API Contracts), 7 (Database)
   - **AI Tasks**: Add Sections 4 (Personas), 5 (Pain Points), 9 (Prompts)
3. **The Specific Task Details**:

```
I'm working on Task [X.X]: [Task Name]

Description: [from task list]
Dependencies: [completed tasks]
Acceptance Criteria: [from task list]
Time Estimate: [from task list]
```

Example Complete Prompt:

```
I'm working on Task 4.1: Create Chat Interface Component
```

[Paste Section 1: Project Overview]
[Paste Section 2: Technical Architecture - Project Structure]
[Paste Section 3: Design System - Colors, Typography, Component Patterns]
[Paste Section 10: Code Standards]

Task Details:
- Create components/genie/chat-interface.tsx
- Add Textarea for user input with submit button
- Show conversation history with user/assistant messages
- Add suggested prompts section
- Handle loading states
- Call /api/analyze-query and /api/generate-ui
- Trigger onUIGenerated callback

Acceptance Criteria:
- Can send messages and see responses
- Loading states are clear
- Follows BevGenie brand colors
- Mobile responsive

Please implement this component.